



SAPIENZA  
UNIVERSITÀ DI ROMA

# Design, Development and Real-world Evaluation of a Distributed Cyber-physical System for Precision Agriculture

Department: Dipartimento di Ingegneria Informatica, Automatica e Gestionale  
Supervisor: Ioannis Chatzigiannakis  
Date and place of submission: 18/May/2018

Djordje Simic

# Abstract

As the world's population continues to grow at unprecedented rates, more and more of the world is becoming urbanised. It is estimated that by the year 2050 the global population will reach 9.6 billion people [25]. At the same time climate changes are already negatively impacting the agricultural production, so farms need to increase the production while preserving the environment. Hydroponic systems provide a potential solution to this problem as they allow for much faster plant growth compared to traditional farming methods, with drastically reduced requirements for water, fertilizers and pesticides. Advances in the field of cheap low-power microcontrollers with long range communication protocols enable fast and cost effective way to automate this systems, thus reducing the need for human labor and providing near perfect conditions for growing plants that can be deployed in both remote and urban environments.

# Table of Contents

<b>1 Introduction</b>	<b>4</b>
1.1 Hydroponics	4
1.1.1 Ebb and Flow method of operation	5
1.2 Wireless Sensor Networks (WSN)	5
1.2.1 Star Topologies	7
1.2.2 Tree Topologies	7
1.2.3 Mesh Topologies	7
1.3 Open hardware	7
<b>2 Low Power Wide Area Networks (LPWAN)</b>	<b>9</b>
2.1 LoRa	10
2.1.1 Sub-GHz operation	12
2.1.2 Modulation method	12
2.1.3 Spreading Factor	14
2.1.4 Bandwidth	14
2.1.5 Data Rate	14
2.2 Network Architecture	15
2.2.1 The Things Network (TTN)	16
2.3 Battery Lifetime	17
2.4 Network Capacity	18
2.5 Device Classes	18
2.6 Security	19
2.7 LoRaWAN Regional Summary	20
2.8 Comparison with other LPWAN protocols	21
<b>3 STM32 NUCLEO Prototyping Board</b>	<b>23</b>
3.1 Cortex-M4	26
3.2 NVIC	27
<b>4 A Smart Hydroponics Solution</b>	<b>29</b>
4.1 LoRa_sensors	33

4.2 LoRa_relays	34
4.3 Hardware	36
4.3.1 DS18B20 Waterproof temperature sensor	36
4.3.2 DHT11 Temperature/Humidity sensor	38
4.3.3 FD45PI10 magnetic float switch	40
4.3.4 RFM96W	41
4.4 Software	41
4.4.1 Node-Red	41
4.4.2 InfluxDB	41
<b>5 Conclusion</b>	<b>42</b>
<b>6 References</b>	<b>43</b>

# 1 Introduction

## 1.1 Hydroponics

Hydroponic systems are a subset of hydroculture, the method of growing plants without soil, using mineral solutions in a water solvent [6] There are many different ways of achieving this, differing in the design of the system, substrates used (growing medium) and sources of nutrients.

Independently of the kind of the system, substrates and nutrients, the general way of hydroponics operation is that the plants are placed in a growing medium and nutrients are provided directly to the roots so the plant has everything it needs readily available, thus there is no wasting of the energy in developing big root systems like soil based systems, resulting in bigger yields and healthier, faster growing plants.

Compared to the traditional ways of growing hydroponics offer various benefits:

- Require 1/5 of the space
- Use around 10% of water compared to soil growing. The run-off water from traditional growing methods can be a potential environment danger since it contains high levels of calcium, phosphorus and potassium, while hydroponics water can be reused multiple times reducing the water consumption even more
- Can be built on infertile land, rooftops, basements, etc.
- Provide sterile grow environment, eliminating the need for pesticides and herbicides
- Complete control of nutrients
- Use around 1/4 of fertilizers
- Greater yields
- Faster grow
- Year-round growing
- Less labor involved

There are also disadvantages:

- Higher initial setup cost
- Need for constant supervision
- Susceptibility to power outages
- Unwanted water-based microorganisms
- Technical knowledge needed

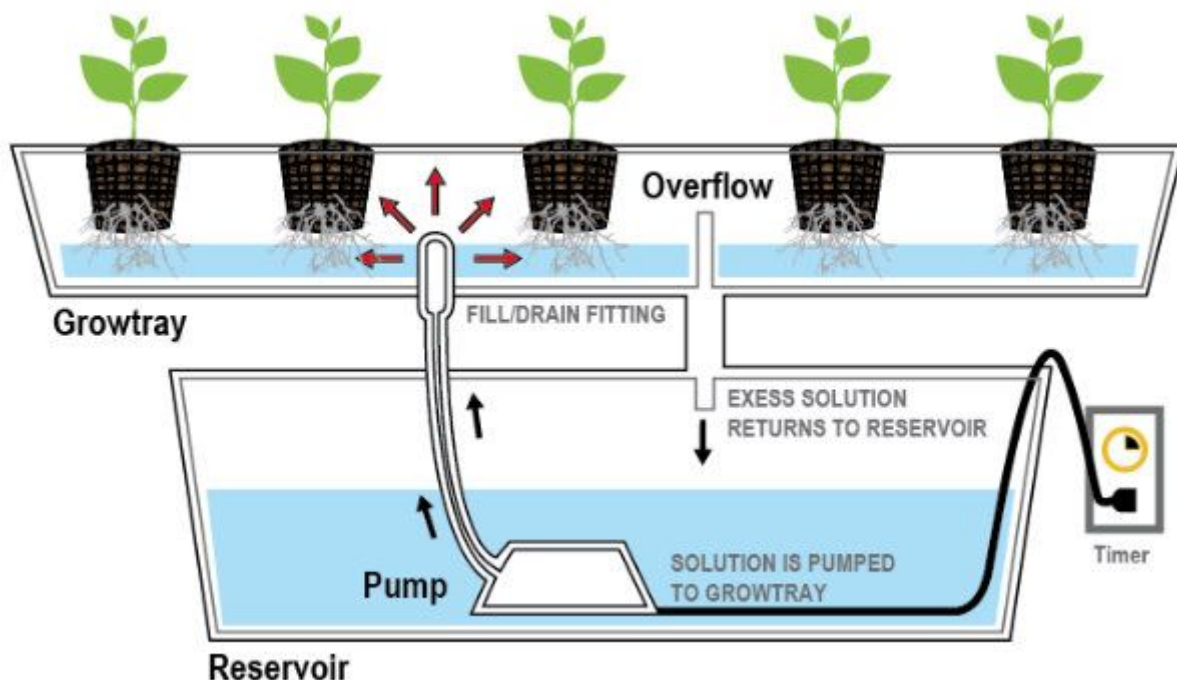
Some of the disadvantages can be addressed by automating processes and introducing constant monitoring and reporting which is the goal of this project.

The configuration used in this project is:

- Ebb and Flow growing technique
- Clay pebbles as substrate

- Liquid three component nutrients

### 1.1.1 Ebb and Flow method of operation



*Figure 1: Ebb and Flow generic setup*

The system consists of two parts: main reservoir and flood tray. Main reservoir holds the nutrient solution which is a mix of pH adjusted water and nutrients (both pH and nutrient ratio depend on the types of the plants grown). Nutrient solution is periodically pumped to the flood tray, usually 3 to 4 times per day in the duration of 15 to 30 minutes.

Flood tray has two pipes fitted, one is used for filling and draining the tray while the other regulates the water level and can be adjusted as needed. The rule of thumb is that the water level should come to the bottom of the growing nets. The tray is filled with growing medium, in this case with clay pebbles.

This mode of operation supplies the plants roots with necessary nutrients while also providing ample amounts of oxygen which is required for healthy and fast growing.

## 1.2 Wireless Sensor Networks (WSN)

Wireless Sensor Network [1,2] refers to groups of dispersed autonomous sensors used to monitor and record physical conditions of the environment and organizing the data in a central location.

WSNs consists of base stations (gateways) and typically a large number of nodes used to monitor various physical or environmental conditions like sound, pressure, temperature, etc.

Sensor nodes are usually very basic in terms of interfaces and components, they consist of:

- Processing unit with limited computational power
- Small amount of memory
- Sensors
- Communication device - typically radio transceivers
- Power source - usually battery

In addition they can contain:

- Energy harvesting modules
- Secondary ASICs
- Secondary communication interface - USB / RS-232

In contrast to sensor nodes, gateways are not as much resource and energy limited and they serve as a bridge between WSN and Local Area Network or Wide Area Network. This allows to store and process the data by devices with more resources, like a remote server. Wireless wide area networks used for low-power devices are known as Low-Power Wide-Area Networks (LPWAN). LoraWAN protocol discussed in later chapters is a type of LPWAN and is built on top of the Lora radio network.

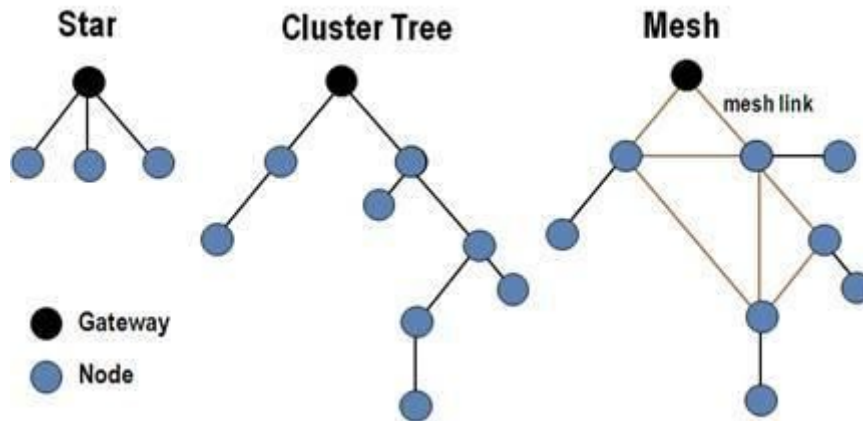
Important factors in LPWAN:

- Network architecture
- Communication range
- Battery lifetime / low power consumption
- Interference robustness
- Maximum number of nodes in a network
- Network security
- One-way vs two-way communication
- Diversity of possible applications

Some of the characteristics of WSNs:

- Node power consumption constraints, since they usually operate using batteries or energy harvesting
- Resilience to node failures
- Node mobility
- Scalability
- Ease of use
- Ability to withstand harsh environmental conditions

Typical topologies used in WSN:



**Figure 2:** Common WSN topologies

### 1.2.1 Star Topologies

In this kind of communication topology nodes are directly connected to the gateway. The gateway can send or receive messages to and from a number of remote nodes, while the nodes cannot communicate between themselves.

This allows for low latency communication while keeping nodes' power consumption at a minimum and enabling simple control, the disadvantage is that the gateway must be in the transmission range of every single node.

### 1.2.2 Tree Topologies

The nodes are organized hierarchically, where each node connects to the one on the level above it, up to the gateway. This enables easy network expansion, however since it is bus-like, if some part breaks the whole network can collapse.

### 1.2.3 Mesh Topologies

This type of topology allows transmission of the data between nodes that are within radio transmission range. If a node wants to send a message to another node which is outside of the radio communication range, it needs an intermediate node that will forward the message to the desired node. This allows for easy isolation and detection of faults, with the downside of the size of the network created and large initial costs.

## 1.3 Open hardware

The concept of open hardware (or open-source hardware) [3] is similar to that of open-source software, meaning that all the design specifications of hardware are licensed in such a way that it can be studied, created, modified and distributed by anyone for free.

Similarly to open-source software code, open hardware blueprints, schematics, logic design, CAD drawings, etc. are available for modification to anyone under permissive licenses. Many open-source hardware projects use existing free open-source software licenses.



Open hardware encourages modification, remixing and reproducing, which is in contrast with closed hardware that relies on obfuscation and patent law, making the recreation and any changes of the objects as hard as possible, enforcing vendor lock-in.

The meaning of “free” in software and hardware is somewhat different, since typically software can be distributed with little or no cost, while the hardware requires physical components and tools to be purchased.

## 2 Low Power Wide Area Networks (LPWAN)

The last decade we witnessed a tremendous progress towards the interconnection of the digital and physical domains, giving rise to the “Internet of Things” (IoT). ICT is increasingly being embedded into the physical world: smartphones, NFC, RFID, and, networked sensors are now common items in our everyday lives. The exponential growth of connected objects that can participate in the IoT ecosystem has already surpassed the number of computers and mobile phones operated by humans. All these devices connected to the Internet enable an ever-growing gamut of application domains and innovative services that change the way we live, work and communicate.

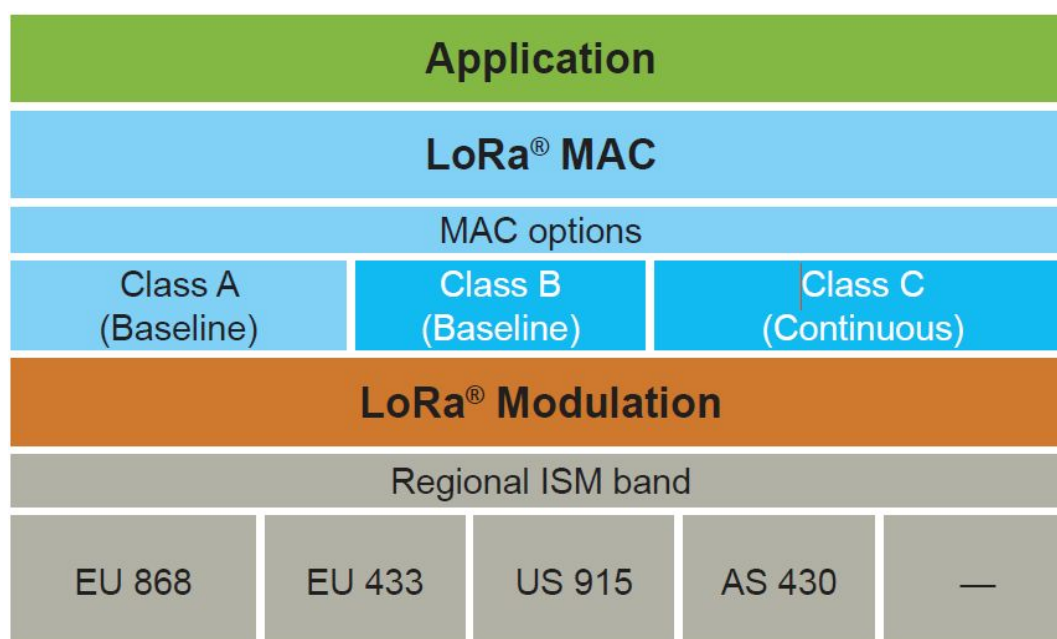
The basis of IoT is the ability to integrate sensing, computation and wireless communication in small, low-power devices that can be seamlessly embedded in complex physical environments. In relation to wireless communications, all the efforts up till now focused predominantly on low-power transmission technologies such as IEEE 802.15.4 (ZigBee, Z-Wave) and IEEE 802.15.11 (Bluetooth, BLE) [11]. These technologies provide reasonably high bit rate exchanges over short-range. Due to this low-power and the short-range mode of operation, deploying large-sized network requires the use of communication protocols that deliver messages based on a multi-path approach [12]. Such a multi-path approach provides certain benefits, such as the capability to overcome communication obstacles [13], and the overall improvement of security of the network [14]. Experimentation over real-world wireless sensor networks has highlighted the difficulties and limitations of the multi-hop short-range paradigm [15]. Several techniques have been proposed to overcome these difficulties, for example, by varying the transmission range of the nodes [16], providing hierarchical network structures [17] or even employing mobile nodes to facilitate network management [18]. Despite all these efforts, the reduced transmission range creates several difficulties that are difficult to overcome. As a result, real-world deployments need to use a combination of networking technologies in order to deliver urban-scale coverage in the context of smart city services [19,20].

New verticals within the Internet of Things (IoT) paradigm such as smart cities, smart farming, or goods monitoring, among many others, are demanding strong requirements to the Radio Access Network (RAN) in terms of coverage, end-node’s power consumption, and scalability. The technologies employed so far to provide IoT scenarios with connectivity, e.g., wireless sensor network and cellular technologies, are not able to simultaneously cope with these three requirements [9]. In the past few years, the approach of exploiting sub-GHz was proposed in order to increase the transmission range of nodes by trading-off data transmission rate while keeping power consumption at low levels [21]. This so-called Low-Power Wide Area Networks (LPWANs) in contrast to more high-frequency communication, low-frequency signals are not as attenuated by thick walls or multipath propagation as high-frequency signals contributing in this way to robustness and reliability of the signal [22]. LPWAN technologies allow IoT devices to connect to Concentrators (also called a *collector* or *concentrator*) over distances in the range of several kilometres. Overall, LPWANs are considered promising candidates for IoT applications, since they allow *high energy autonomy* of the connected devices, *low device and deployment costs*, *high coverage capabilities* and support *large number of devices* [23]. Concretely, the Long-Range

Wide Area Network (LoRaWAN) technology is one of the LP-WAN platforms that is receiving greater attention from both the industry and the academia.

In LPWAN, IoT devices are characterized by high sensitivities, rather than high data rates, are connected in a star topology where all nodes have a direct connection to a central unit. The central unit is called the concentrator and acts as protocol converters and direct the data sent from the end devices to a Network Server (over for example Ethernet or 3G/4G/5G) that manages all the decoding of the packets and handles redundant transmissions. Each message transmitted by the IoT devices provides an Application Server identifier which is used by the Network Server in order to forward the message to the corresponding endpoint. Recently under the fog computing proposal, concentrators are used to offload parts of the logic from the Network Server and the Application Server to the concentrators in order to create offline compatibilities and reduce latencies [24].

LoRaWAN™ [4] defines the communication protocol and system architecture for the network while the LoRa® physical layer enables the long-range communication link. The protocol and network architecture have the most influence in determining the battery lifetime of a node, the network capacity, the quality of service, the security, and the variety of applications served by the network.



**Figure 3:** LoRaWAN protocol stack

## 2.1 LoRa

LoRa® is the physical layer or the wireless modulation utilized to create the long range communication link. Many legacy wireless systems use frequency shifting keying (FSK) modulation as the physical layer because it is a very efficient modulation for achieving low power. LoRa® is based on chirp spread spectrum modulation, which maintains the same low power characteristics as FSK modulation but significantly increases the communication range. Chirp spread spectrum has been used in military and space communication for







decades due to the long communication distances that can be achieved and robustness to interference, but LoRa® is the first low cost implementation for commercial usage.

The advantage of LoRa® is in the technology's long range capability. A single gateway or base station can cover entire cities or hundreds of square kilometers. Range highly depends on the environment or obstructions in a given location, but LoRa® and LoRaWAN™ have a link budget greater than any other standardized communication technology. The link budget, typically given in decibels (dB), is the primary factor in determining the range in a given environment. Below are the coverage maps from the Proximus network deployed in Belgium. With a minimal amount of infrastructure, entire countries can easily be covered.



**Figure 4:** Proximus network coverage in Belgium

However, there isn't a single solution to all IoT requirements. LPWAN positions itself in the range of applications that need multi-year battery lifetime, long range of communication and small packet size that are sent periodically (few times per hour).

	<b>Local Area Network</b> Short Range Communication	<b>Low Power Wide Area</b> (LPWAN) Internet of Things	<b>Cellular Network</b> Traditional M2M
	<b>40%</b>	<b>45%</b>	<b>15%</b>
	Well established standards In building	Low power consumption Low cost Positioning	Existing coverage High data rate
	Battery Live Provisioning Network cost & dependencies	High data rate Emerging standards	Autonomy Total cost of ownership
	Bluetooth 4.2 	LoRa® 	 3G+ / H+ 4G

**Figure 5:** Comparison of various network types

### 2.1.1 Sub-GHz operation

LoRa operates in sub-GHz range (867-869 MHz in Europe), the preferred mode of operation in the power-constrained design. This comes from the fact that the lower the frequency, the less power is required to maintain a link budget at a specified range, shown by the Friis transmission equation:

$$P_r = \frac{P_t G_t G_r M \lambda^2}{(4\pi)^2 d^2}$$

Where:

$P_t$  = transmitted power

$P_r$  = received power

$G_t$  = transmitter antenna gain

$G_r$  = receiver antenna gain

$\lambda$  = wavelength

$d$  = distance between transmitter and receiver

Low frequency transmissions imply lower data rates but this doesn't pose a problem since typically IoT applications don't have high throughput requirements. Another advantage of using lower data rates is the reduced error rate, allowing for a decrease in receiver sensitivity.

However this is not without some downsides, one of which is that slow links increase the duty-cycle, allowing for more interference from noise and other signals. Additionally, longer message transmit time implies increased power consumption on both transmitter and receiver ends.

### 2.1.2 Modulation method

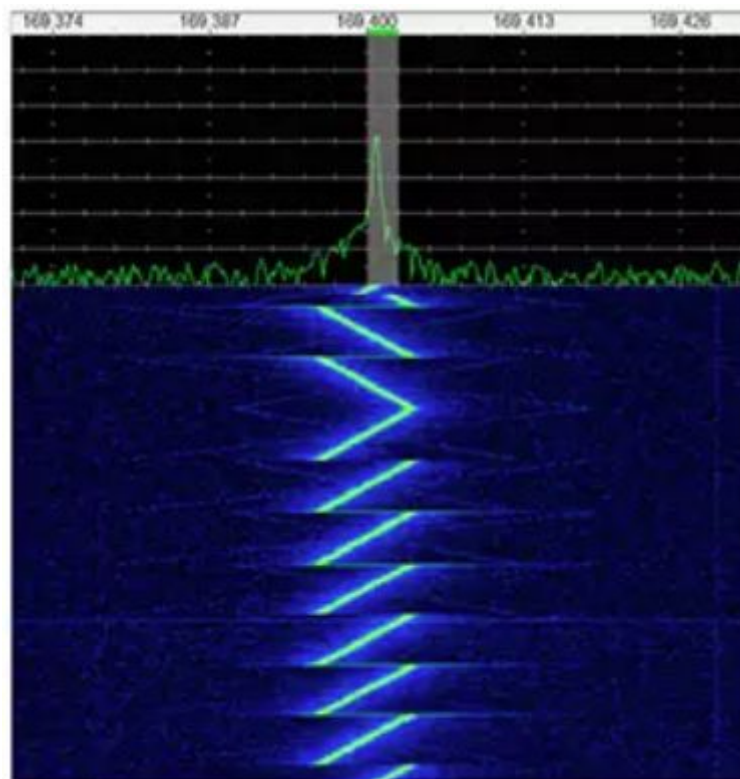
LoRa utilizes a variation of spread spectrum modulation, a technique originally developed for the military and later accepted by other industries because of its security, resilience to noise and high data rates.

Spread spectrum technique works by dispersing ("spreading") a relatively small data signal over the whole carrier frequency, which can in some cases be 1000 times broader than the common narrow band (25 kHz information bandwidth compared with up to 26 MHz for spread spectrum).

Although this technique supports high data rates, it requires high bandwidth carriers and advanced modulation/demodulation of the signal, increasing design complexity and power requirements. Since IoT applications usually don't have the need for high data rates, traditional spread spectrum techniques are not suitable for them.

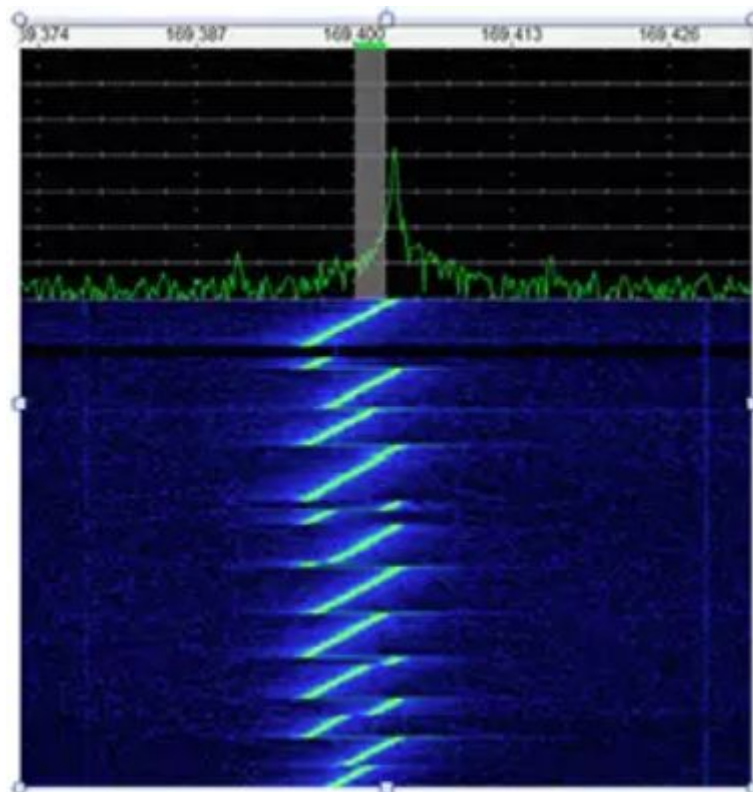
LoRa utilizes a unique chirp spread spectrum modulation, which uses frequency chirps with a linear variation of frequency over time to encode information. Since chirp pulses are linear, they are easy to eliminate in the decoder, while also adding the benefit of resistance to Doppler effect and reducing the price of the transmitters (crystals in them don't need to have extreme accuracy). This brings some of the benefits of the spread spectrum noise immunity, while simplifying design requirements and lowering the cost.

Chirps can be generated by a relatively simple fractional-N phase locked loop (PLL). When initiating a transmission, a LoRa modem first sends a preamble that is comprised of a series of chirps and ends with a "reverse chirp", as seen on the image below.



**Figure 6:** LoRa preamble (newest data at the top). Source: [5]

The receiver "locks" onto the preamble signal and starts listening. Transmission continues with the actual payload which has a series of symbols that function similarly to multiple frequency tones in M-ary Frequency Key Shifting.



**Figure 7:** LoRa data modulation. Source: [5]

Another powerful feature of LoRa modulation is that the modem can simultaneously receive data from multiple transmitters on the same frequency, provided that they use different chirp rate, called “spreading factor”.

### 2.1.3 Spreading Factor

A spreading factor is the duration of the chirp. LoRa operates with spreading factors from 7 to 12, SF7 being the shortest time on air, SF12 the longest. Each consecutive step in the spreading factor doubles the time on air, resulting in less data transmitted per unit of time.

### 2.1.4 Bandwidth

LoRa uses three bandwidths: 125kHz, 250kHz and 500kHz. The chirp uses the entire bandwidth.

### 2.1.5 Data Rate

Data rates are different configurations of frequencies, spreading factors and bandwidths, depending on the location where LoRa will be utilized. Below is the table of data rates for regions EU433, EU868, CN780 and AS923.



Data Rate	Configuration	bits/s	Max payload
DR0	SF12/125kHz	250	59
DR1	SF11/125kHz	440	59
DR2	SF10/125kHz	980	59
DR3	SF9/125kHz	1 760	123
DR4	SF8/125kHz	3 125	230
DR5	SF7/125kHz	5 470	230
DR6	SF7/250kHz	11 000	230
DR7	FSK: 50kbps	50 000	230

**Figure 8:** LoRa data rates

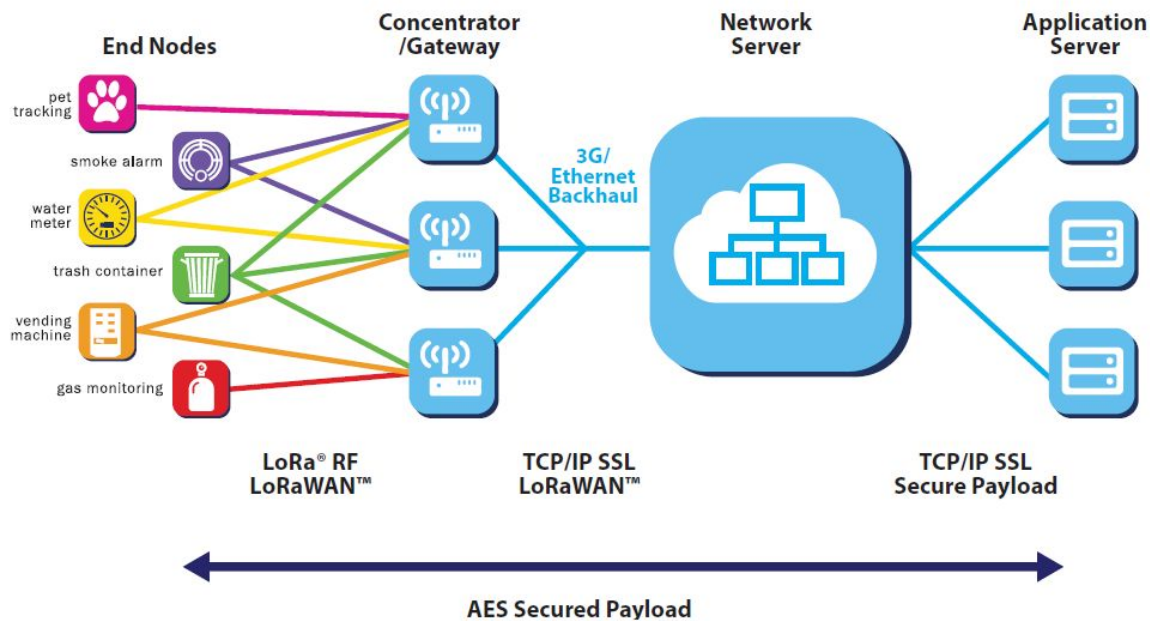
DR	Urban Scenario	Suburban Scenario	Rural Scenario
DR0	6.5	6.7	18.5
DR1	5.4	5.8	13.7
DR2	4.5	5.7	11.6
DR3	4.4	5.3	11.5
DR4	4	5.0	10.3
DR5	2.8	5.5	9.6

**Figure 9:** Longest range (km) obtained in empirical testing of different data rates in various scenarios.  
Source: [9]

## 2.2 Network Architecture

LoRa is just the physical layer (PHY) mechanism. In IoT solutions developers need the complete network protocol stack that will build on top of the LoRa PHY. This is what LoRaWAN standard does, as it provides the definition of the Media Access Control (MAC) layer designed to operate with the LoRa PHY [4].





**Figure 10: LoRaWAN network architecture**

LoRaWAN utilizes star topology, discussed in *Section 1.2.1*.

In a LoRaWAN™ network nodes are not associated with a specific gateway. Instead, data transmitted by a node is typically received by multiple gateways. Each gateway will forward the received packet from the end-node to the cloud-based network server via some backhaul (either cellular, Ethernet, satellite, or Wi-Fi).

There are two types of gateways, full and single-channel. Former support listening on up to 8 channels simultaneously while the latter can listen on only one, however they are much cheaper for deployment, therefore they are widely used in testing and prototyping.

The intelligence and complexity is pushed to the network server, which manages the network and filters redundant received packets (deduplication), performs security checks, schedules acknowledgments through the optimal gateway, performs adaptive data rate, etc. If a node is mobile or moving there is no handover needed from gateway to gateway, which is a critical feature to enable asset tracking applications – a major target application vertical for IoT. In this project these functionalities are performed by The Things Network, described below.

### 2.2.1 The Things Network (TTN)

TTN [7] is an open source project aiming to provide global and crowd sourced Internet of Things network. The Things Network provides a LoRaWAN Network Server. LoRaWAN is a “network-intensive” protocol, intensive in the sense that due to the simple and minimalistic approach for devices, the backend systems (also called Network Servers) are responsible for most of the logic. LoRaWAN was designed for the centralized architecture of telecom

operators, so in order to run on a distributed infrastructure like The Things Network, some steps had to be added. In The Things Network's backend there is a number of different core functions.

Firstly, there are Gateway-related functions such as scheduling and managing the utilization of the gateways. Scheduling is needed because a gateway can only do one transmission at the same time. The utilization information is used to evenly distribute load over different gateways and to be compliant with the European duty cycles. Another important feature is monitoring the status of each gateway.

Secondly, device-related functions that manage the state of devices in the network are needed. As device address are non-unique, the network has to keep track of which addresses are used by which devices in order to map a message to the correct device and application. Other things the network must keep track of are the security keys and frame counters.

Thirdly there is some functionality related to applications. For example, the Brokers and Handlers need to know to which server traffic for a specific application needs to be forwarded. The Handlers need to know how to interpret binary data, and bridge to higher-layer protocols, such as AMQP and MQTT.

Finally, and most importantly, as The Things Network is a distributed network, there has to be functionality that supports this distribution. Service discovery functionality helps components to determine where traffic should be routed to. Currently, this is implemented as a centralized Discovery server, giving The Things Network Foundation control over which components are allowed to announce specific services.

Limitations are imposed on the amount of messages sent by the nodes, particularly by duty cycle regulations (this depends on the region, in Europe it is set to 1% for frequencies from 868 to 868.6 MHz) and the Fair Access Policy from TTN public community network, which limits the uplink airtime to 30 seconds and 10 downlink messages per day. The reason for a low limit on downlink comes from the fact that the gateways currently don't support full duplex communication, meaning that while a single downlink message is transmitted, up to 8 uplink messages could be lost.

TTN provides a web interface where new gateways, applications and devices can be registered. Then, necessary keys are generated and copied to configuration files on the end nodes, this way the communication is encrypted end-to-end.

## 2.3 Battery Lifetime

The nodes in a LoRaWAN™ network are asynchronous and communicate when they have data ready to send whether event-driven or scheduled. This type of protocol is typically referred to as the Aloha method. In a mesh network or with a synchronous network, such as cellular, the nodes frequently have to 'wake up' to synchronize with the network and check for messages. This synchronization consumes significant energy and is the number one driver of battery lifetime reduction. In a recent study and comparison done by GSMA of the

various technologies addressing the LPWAN space, LoRaWAN™ showed a 3 to 5 times advantage compared to all other technology options.

## 2.4 Network Capacity

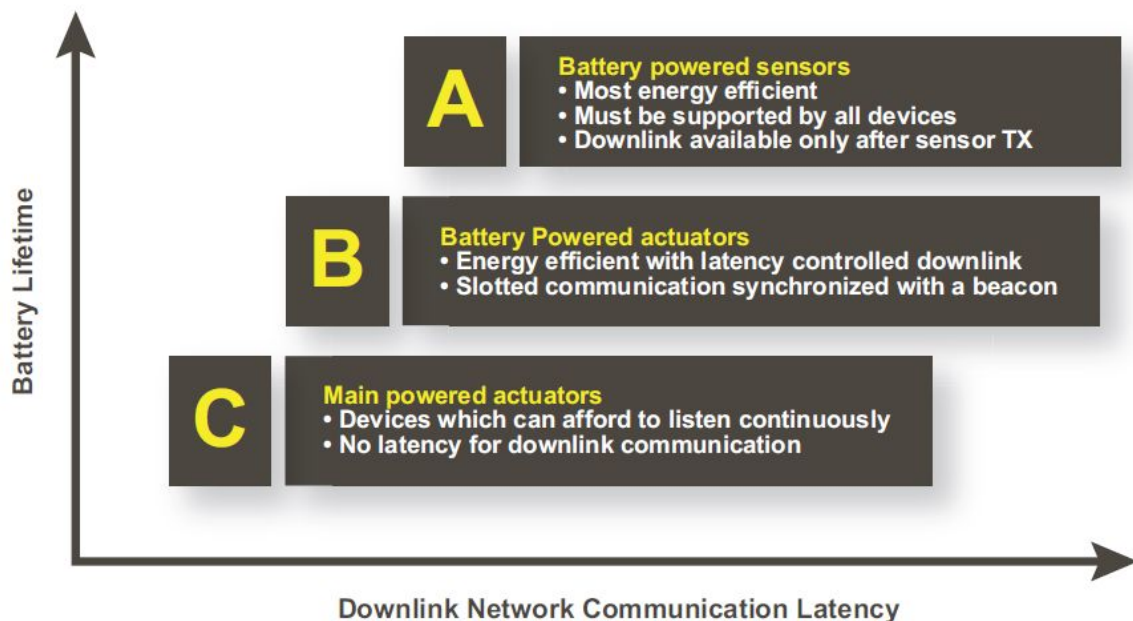
In order to make a long range star network viable, the gateway must have a very high capacity or capability to receive messages from a very high volume of nodes. High network capacity in a LoRaWAN™ network is achieved by utilizing adaptive data rate and by using a multichannel multi-modem transceiver in the gateway so that simultaneous messages on multiple channels can be received. The critical factors affecting capacity are the number of concurrent channels, data rate (time on air), the payload length, and how often nodes transmit. Since LoRa® is a spread spectrum based modulation, the signals are practically orthogonal to each other when different spreading factors are utilized. As the spreading factor changes, the effective data rate also changes. The gateway takes advantage of this property by being able to receive multiple different data rates on the same channel at the same time. If a node has a good link and is close to a gateway, there is no reason for it to always use the lowest data rate and fill up the available spectrum longer than it needs to. By shifting the data rate higher, the time on air is shortened opening up more potential space for other nodes to transmit. Adaptive data rate also optimizes the battery lifetime of a node. In order to make adaptive data rate work, symmetrical uplink and downlink is required with sufficient downlink capacity. These features enable a LoRaWAN™ network to have a very high capacity and make the network scalable. A network can be deployed with a minimal amount of infrastructure, and as capacity is needed, more gateways can be added, shifting up the data rates, reducing the amount of overhearing to other gateways, and scaling the capacity by 6-8x. Other LPWAN alternatives do not have the scalability of LoRaWAN™ due to technology trade-offs, which limit downlink capacity or make the downlink range asymmetrical to the uplink range.

## 2.5 Device Classes

End-devices serve different applications and have different requirements. In order to optimize a variety of end application profiles, LoRaWAN™ utilizes different device classes. The device classes trade off network downlink communication latency versus battery lifetime. In a control or actuator-type application, the downlink communication latency is an important factor.

Bi-directional end-devices (Class A): End-devices of Class A allow for bi-directional communications whereby each end-device's uplink transmission is followed by two short downlink receive windows. The transmission slot scheduled by the end-device is based on its own communication needs with a small variation based on a random time basis (ALOHA-type of protocol). This Class A operation is the lowest power end-device system for applications that only require downlink communication from the server shortly after the end-device has sent an uplink transmission. Downlink communications from the server at any other time will have to wait until the next scheduled uplink. Bi-directional end-devices with scheduled receive slots (Class B): In addition to the Class A random receive windows, Class B devices open extra receive windows at scheduled times. In order for the end-device

to open its receive window at the scheduled time, it receives a time-synchronized beacon from the gateway. This allows the server to know when the end-device is listening. Bi-directional end-devices with maximal receive slots (Class C): End-devices of Class C have almost continuously open receive windows, only closed when transmitting.



**Figure 11:** LoRa device classes comparison

## 2.6 Security

It is extremely important for any LPWAN to incorporate security. LoRaWAN™ utilizes two layers of security: one for the network and one for the application. The network security ensures authenticity of the node in the network while the application layer of security ensures the network operator does not have access to the end user's application data. AES encryption is used with the key exchange utilizing an IEEE EUI64 identifier. There are trade-offs in every technology choice but the LoRaWAN™ features in network architecture, device classes, security, scalability for capacity, and optimization for mobility address the widest variety of potential IoT applications.

LoRaWAN 1.0 specifies a number of security keys: NwkSKey, AppSKey and AppKey. All keys have a length of 128 bits [8].

The network session key (NwkSKey) is used for interaction between the Node and the Network. This key is used to check the validity of messages (MIC check). In the backend of The Things Network this validation is also used to map a non-unique device address (DevAddr) to a unique DevEUI and AppEUI.

The application session key (AppSKey) is used for encryption and decryption of the payload. The payload is fully encrypted between the Node and the Handler component of The Things Network.

These two session keys (NwkSKey and AppSKey) are unique per device, per session. If a device is dynamically activated (OTAA), these keys are re-generated on every activation. If it is statically activated (ABP), these keys stay the same until changed by the user.

Dynamically activated devices (OTAA) use the application key (AppKey) to derive the two session keys during the activation procedure. In The Things Network a default AppKey can be used for activation of all devices, or it can be customized per device.

Because LoRa is a radio protocol, anyone is able to capture and store messages. It is not possible to read these messages without the AppSKey, because they are encrypted, nor is it possible to tamper with them without the NwkSKey, because this will make the MIC check fail. However, it is possible to re-transmit the messages. These so-called replay attacks can be detected and blocked using frame counters.

When a device is activated, these frame counters (FCntUp and FCntDown) are both set to 0. Every time the device transmits an uplink message, the FCntUp is incremented and every time the network sends a downlink message, the FCntDown is incremented. If either the device or the network receives a message with a frame counter that is lower than the last one, the message is ignored.

## 2.7 LoRaWAN Regional Summary

The LoRaWAN™ specification varies slightly from region to region based on the different regional spectrum allocations and regulatory requirements. The LoRaWAN™ specification for Europe and North America are defined, but other regions are still being defined by the technical committee. Joining the LoRa® Alliance as a contributor member and participating in the technical committee can have significant advantages to companies targeting solutions for the Asia market.

	Europe	North America	China	Korea	Japan	India
Frequency band	867-869MHz	902-928MHz	470-510MHz	920-925MHz	920-925MHz	865-867MHz
Channels	10	64 + 8 + 8	In definition by Technical Committee	In definition by Technical Committee	In definition by Technical Committee	In definition by Technical Committee
Channel BW Up	125/250kHz	125/500kHz				
Channel BW Dn	125kHz	500kHz				
TX Power Up	+14dBm	+20dBm typ (+30dBm allowed)				
TX Power Dn	+14dBm	+27dBm				
SF Up	7-12	7-10				
Data rate	250bps- 50kbps	980bps-21.9kbps				
Link Budget Up	155dB	154dB				
Link Budget Dn	155dB	157dB				

**Figure 12:** Communication specifications by regions

## 2.8 Comparison with other LPWAN protocols

LPWANs are going through expansion right now in the IoT sector, the main reason being that there is a strong business incentive considering the relatively low cost of deployment in unlicensed bands (only a fraction compared to cellular).

However there are still some questions to be answered in order to compare different LPWAN technologies:

- Flexibility in supporting wide range of applications
- Security
- Technical aspects - capacity, range, support for two way communication, robustness to interference
- Cost of network deployment, cost of end node BOM, cost of battery (largest part of BOM)
- Capability of the ecosystem to ensure quality and longevity of the solution

Feature	LoRaWAN	Narrow-Band	LTE Cat-1 2016 (Rel12)	LTE Cat-M 2018 (Rel13)	NB-LTE 2019(Rel13+)
Modulation	SS Chirp	UNB / GFSK/BPSK	OFDMA	OFDMA	OFDMA
Rx bandwidth	500 - 125 KHz	100 Hz	20 MHz	20 - 1.4 MHz	200 KHz
Data Rate	290bps - 50Kbps	100 bit/sec 12 / 8 bytes Max	10 Mbit/sec	200kbps – 1Mbps	~20K bit/sec
Max. # Msgs/day	Unlimited	UL: 140 msgs/day	Unlimited	Unlimited	Unlimited
Max Output Power	20 dBm	20 dBm	23 - 46 dBm	23/30 dBm	20 dBm
Link Budget	154 dB	151 dB	130 dB+	146 dB	150 dB
Battery lifetime - 2000mAh	105 months	90 months		18 months	
Power Efficiency	Very High	Very High	Low	Medium	Med high
Interference immunity	Very high	Low	Medium	Medium	Low
Coexistence	Yes	No	Yes	Yes	No
Security	Yes	No	Yes	Yes	Yes
Mobility / localization	Yes	Limited mobility, No loc	Mobility	Mobility	Limited Mobility No Loc

**Figure 13: LoRaWAN side-by-side comparison with Narrow-Band, LTE Cat-1, LTE Cat-M, NB-LTE**

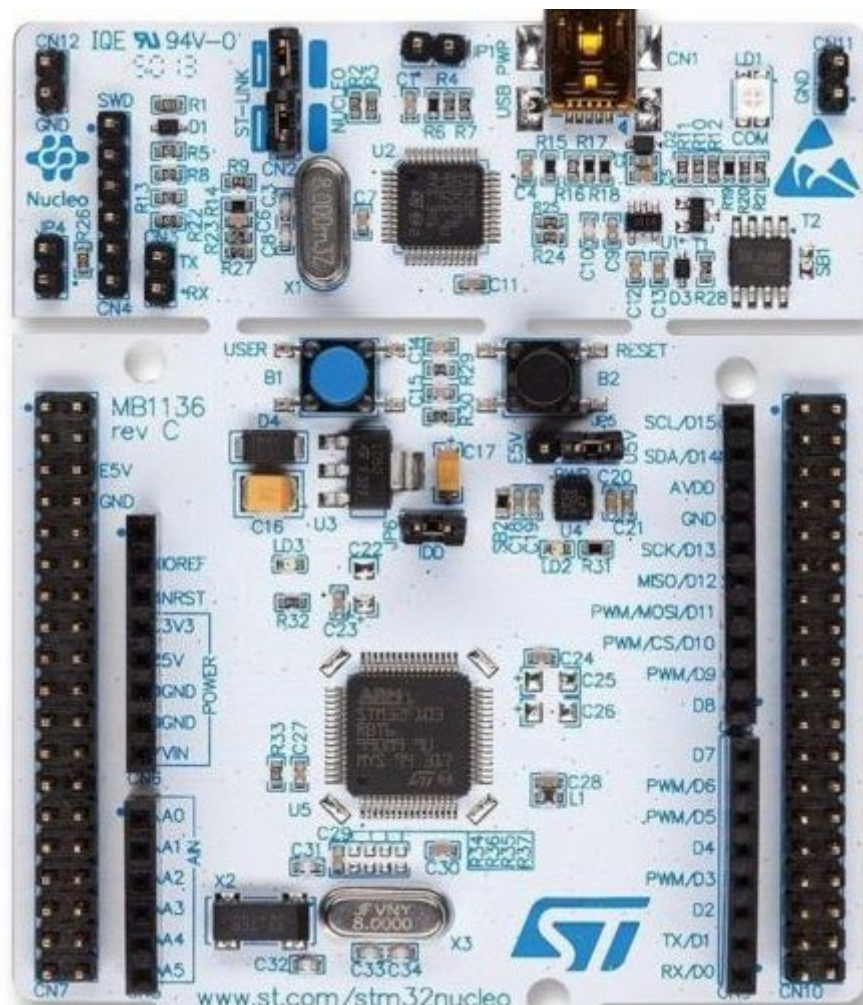


### 3 STM32 NUCLEO Prototyping Board

STM32 Nucleo is a development platform produced by STMicroelectronics [10]. This range of products is aimed at rapid prototyping, trying out new ideas and easy code portability across the STM32 family. All the boards share the same connectors and pin configurations, providing also an integrated ST-Link debugger/programmer, eliminating the need for a separate probe.

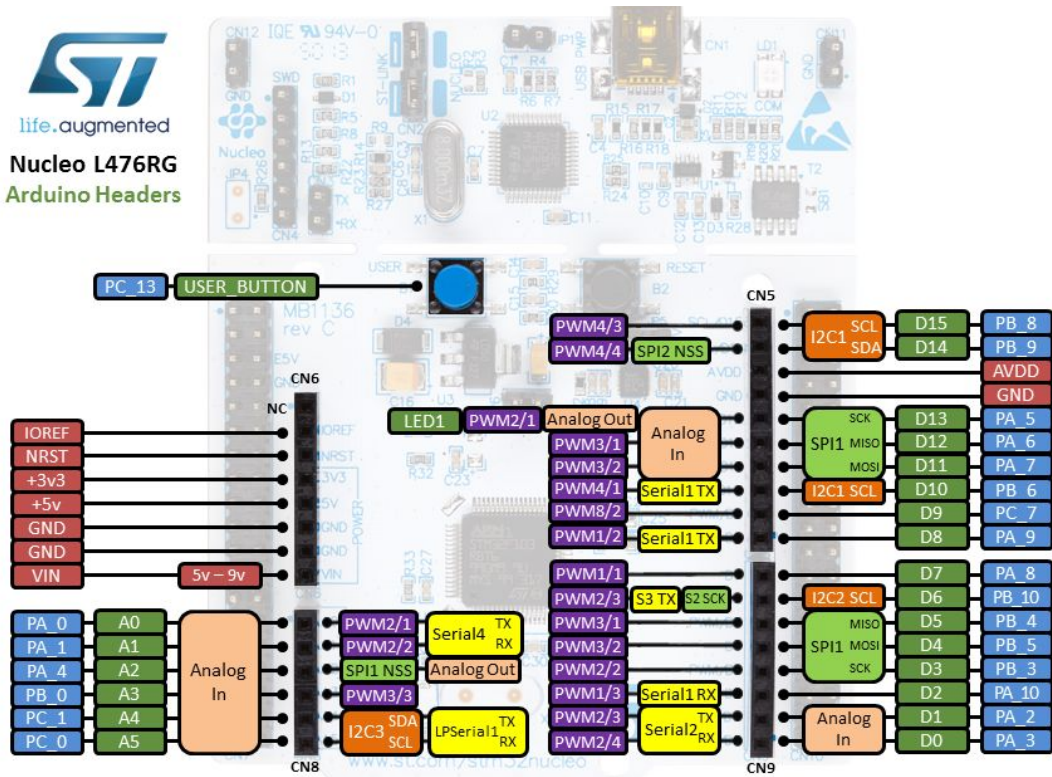
ST provides an STM32 software Hardware Abstraction Layer library (HAL), designed as a driver layer that exposes a set of APIs that interact with the upper layers (application, libraries and stacks), eliminating the need for those layers to know the specifics of the underlying MCU. This allows for improved code reusability and guarantees easy portability to other devices.

The MCU used in this project is STM32L476RG, chosen for its ultra-low-power features. It is based on an ARM Cortex-M4 32-bit RISC core operating at frequencies of up to 80 MHz.



**Figure 14:** STM32L476RG Nucleo



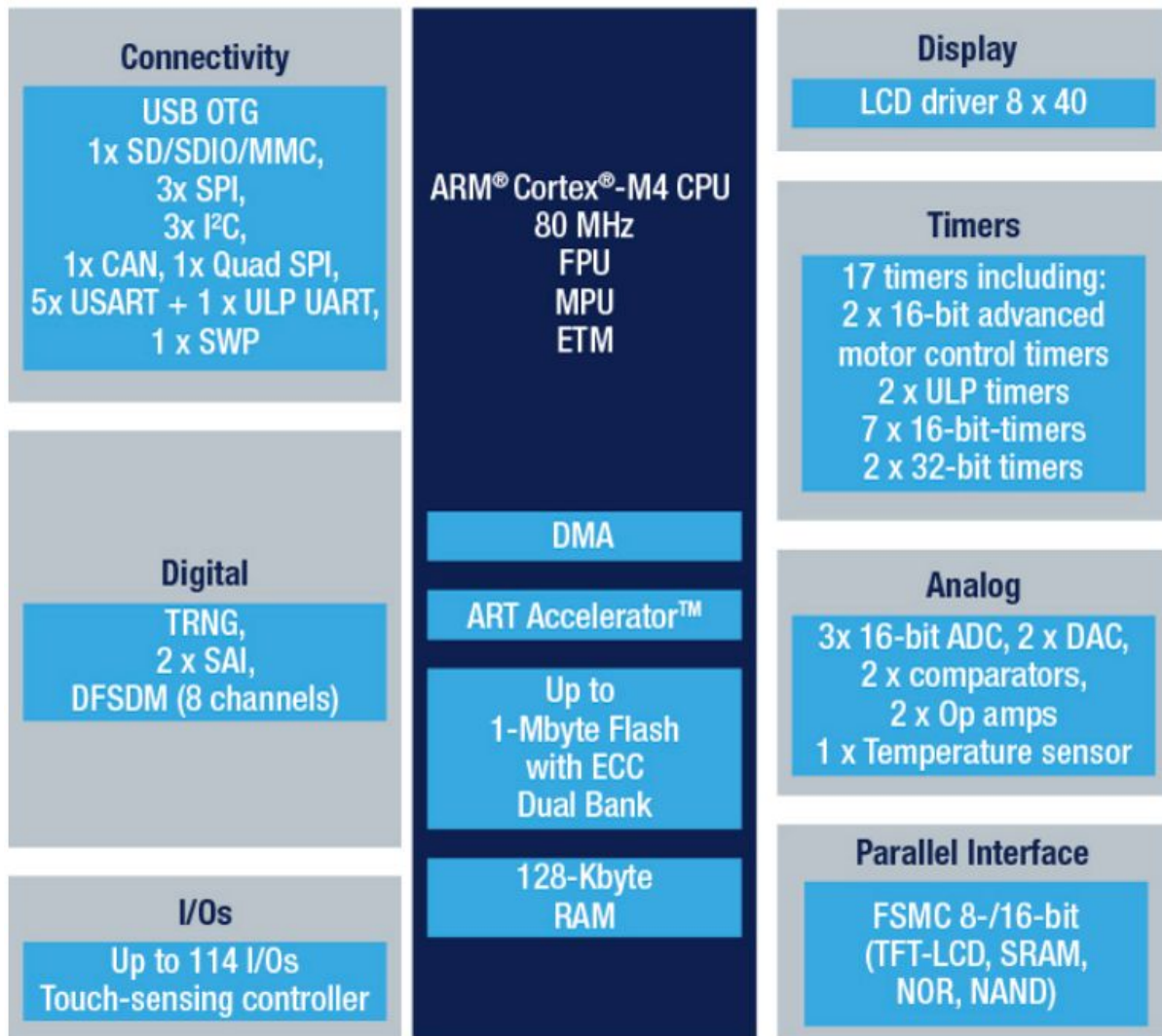


**Figure 15:** STM32L476 Arduino connectors mapping. Source: [10]

#### Key features:

- STM32 microcontroller in LQFP64 package
- Two types of extensions:
  - Arduino Uno V3 connectivity
  - ST morpho extension pin headers
- On-board ST-LINK/V2-1 debugger and programmer
- Three LEDs:
  - USB communication (LD1)
  - user LED (LD2)
  - power LED (LD3)
- Two push-buttons: USER and RESET
- Peripherals:
  - 64 x GPIO: User configurable general purpose I/O ports
  - 3 x USART
  - 3 x I2C
  - 1 x RTC: Real-Time Clock unit used for keeping the time/calendar, providing also alarm interrupts
  - 16 x Timers: SysTick, low-power, generic
- Clock sources:
  - HSE: High Speed External crystal oscillator (4-48 MHz)
  - HSI16: High Speed Internal (16 MHz)
  - LSI: Low Speed Internal, low power (32 kHz)
  - LSE: Low Speed External, used for RTC (32 kHz)

- MSI: Multi Speed Internal (100 kHz - 48 MHz)
- PLL: Phase-Locked Loop that can be used with MSI, HSE or HSI16 (up to 80 MHz)
- Ultra-low-power
  - 1.71 V to 3.6 V power supply
  - -40 °C to 85/105/125 °C temperature range
  - 300 nA in VBAT mode: supply for RTC and 32x32-bit backup registers
  - 30 nA Shutdown mode (5 wake up pins)
  - 120 nA Standby mode (5 wake up pins)
  - 420 nA Standby mode with RTC
  - 1.1 µA Stop 2 mode, 1.4 µA with RTC
  - 100 µA/MHz run mode (LDO Mode)
  - 39 µA/MHz run mode (@3.3 V SMPS Mode)
  - 4 µs wakeup from Stop mode
  - Brown out reset (BOR)
- Memories
  - Up to 1 MB Flash, 2 banks read-while-write, proprietary code readout protection
  - Up to 128 KB of SRAM including 32 KB with hardware parity check
  - External memory interface for static memories supporting SRAM, PSRAM, NOR and NAND memories
  - Quad SPI memory interface
- Development support: serial wire debug (SWD), JTAG, Embedded Trace Macrocell



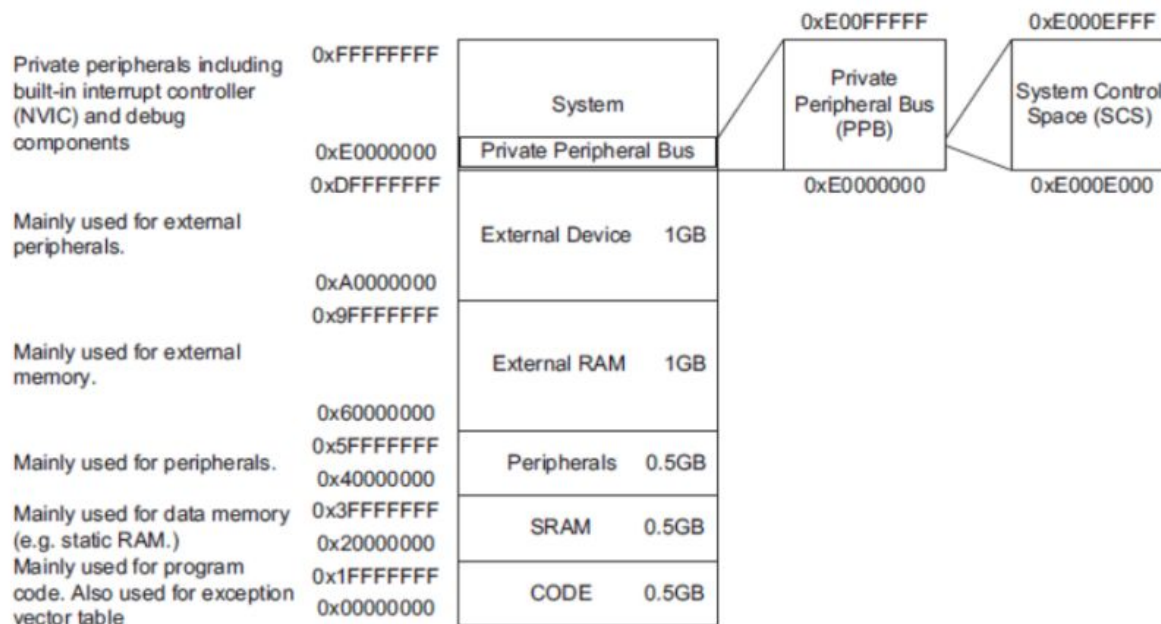
**Figure 16:** STM32L476 circuit diagram. Source: [10]

### 3.1 Cortex-M4

The Cortex-M4 is a part of the Cortex-M family made by ARM, it is a 32-bit RISC processor based on the Harvard architecture. This line of processors offers a balance between low power consumption and performance, while Cortex-M0/M0+/M23 have low power features and Cortex-M7 is high performance. Cortex-M4 has a Floating Point Unit (FPU) which supports all ARM single-precision data-processing instructions and data types. Digital Signal Processor (DSP) instructions are also supported, along with a memory protection unit (MPU) used for enhanced application security.

Cortex-M4 has 13 general purpose registers, marked R0-R12, with special registers for Stack Pointer (R13), Subroutine Link register (R14, used for exception handling), Program Counter (R15) and a Current Program Status Register (R16, handles several flags like zero flag, negative, etc.).

Memory space is standardized across the Cortex-M family in order to ensure code portability. Cortex-M4 has up to 4GB of memory, organized with respect to logical functionalities.



**Figure 17: Cortex-M4 memory mapping**

Cortex-M4 uses the Thumb-2 instruction set which provides smaller code size compared to the 32-bit instruction set (about 25%) while offering the same performance. This is achieved by enriching the limited 16-bit instruction set of Thumb with additional 32-bit instructions, thus producing a variable length instruction set.

One of the most important features of ARM architecture is the interrupt and exception management. When some of those two events occur, the current task is suspended and using the Stack Pointer its context is saved, then a corresponding action is taken, either an exception handler is called or an Interrupt Service Routine. ISRs are managed by the Nested Vectored Interrupt Controller (NVIC), described below.

## 3.2 NVIC

ARM Cortex-M family uses a Nested Vectored Interrupt Controller (NVIC). Vectored is referred to the fact that a vector table is used for organizing interrupts.

32-bit entries point to the corresponding Interrupt Service Routines (ISR). Exception numbers from 1 to 15 are ARM-specific, while the numbers above 15 are vendor specific, thus IRQ numbers from -1 to -14 are defined by the ARM core, while everything greater than 0 is vendor specific (typically for devices like UART/I2C, etc.)

Interrupt priorities are expressed through a 8-bit priority register, the number of bits implemented is left to be decided by the vendor, ARM specifies that for Cortex-M3/M4/M7 minimum is 3 bits.

In this project interrupt is enabled on the pin *PA7* to use it with the float switch. It is configured from the STM32CubeMX.

Exception number	IRQ number	Offset	Vector
255	239	0x03FC	IRQ239
.	.	.	.
18	2	0x004C	IRQ2
17	1	0x0048	IRQ1
16	0	0x0044	IRQ0
15	-1	0x0040	Systick
14	-2	0x003C	PendSV
13		0x0038	Reserved
12			Reserved for Debug
11	-5	0x002C	SVCall
10			Reserved
9			
8			
7			
6	-10	0x0018	Usage fault
5	-11	0x0014	Bus fault
4	-12	0x0010	Memory management fault
3	-13	0x000C	Hard fault
2	-14	0x0008	NMI
1		0x0004	Reset
		0x0000	Initial SP value

**Figure 18: ARM Cortex-M4/M7 vector table**

Enabling interrupts is done in three steps:

1. (optional): Set the priority level of the interrupt
2. Enable the interrupt inside the device
3. Enable the interrupt in the NVIC

ARM Cortex-M uses an interrupt priority numbering in which low numbers are used for high interrupt priorities. *Reset*, *Non-Maskable Interrupt* and *HardFault* are assigned priorities -3, -2 and -1 respectively, others are user defined.

## 4 A Smart Hydroponics Solution

Research was conducted on the topics of hydroponics, low-power networks, low-power MCUs, sensors that should be used and the overall architecture of the system.

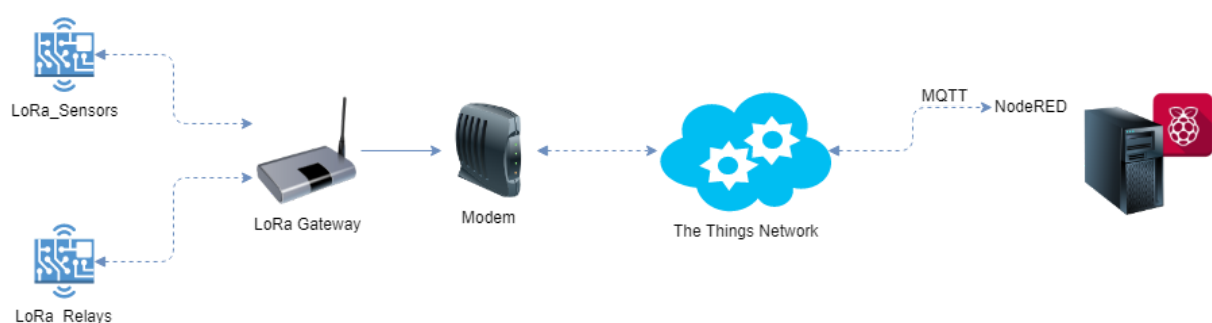
After settling on the type of the hydroponic system that is going to be used (Ebb and Flow), a list of requirements for monitoring and actuation were made. These include:

- Temperature monitoring (ambient and nutrient solution)
- Humidity monitoring
- Water level in the main tank
- Actuation of growing lights and water pump

Considering that the system was planned to be modular and possibly used in scenarios other than hydroponics (outdoor sensing in remote environments), a decision was made to use two modules, one battery powered containing sensors and the other one used for actuation of high current devices and powered from the grid.

Since the requirements for the modules are different, also their mode of operation slightly differs. The sensors module is by default in a low-power sleep mode, periodically waking up to read the sensor data and send it to the server, followed by immediately going back to sleep, the only exception to this is that it can be woken up by the interrupt generated by the water level sensor.

The relays module is always on, there are no power consumption requirements (it is powered from the mains), also it needs to be able to receive messages in real time since the user can manually toggle the device's on/off state.



**Figure 19:** Overall system architecture

The two modules send/receive packets to/from the LoRa gateway which forwards them over the internet to The Things Network server. The server parses the messages, converts them to JSON and publishes them to an MQTT topic. Raspberry Pi server runs an instance of NodeRED which is subscribed to the TTN MQTT topic.



There are three types of messages that can be exchanged:

- Configuration (server → end\_node) - set sensor read rate / set relay on/off schedule
- Sensor read (end\_node → server) - read all sensor values according to the predefined period, send all of the values in one message
- Relay write (server → end\_node) - override predefined schedule and turn the relay on/off manually

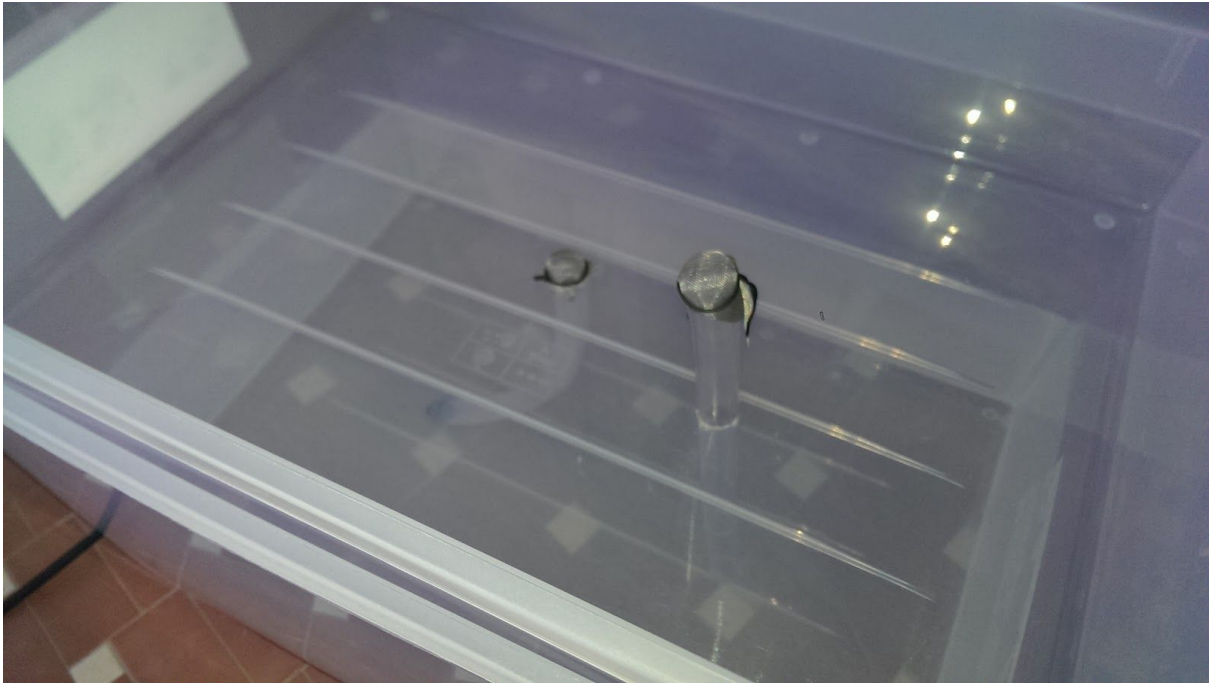
Upon receiving the sensor readings message, server stores the values in InfluxDB (more in *Section 4.4.2*). This data is available visually to the user via a web app.

After research and prototype development phase, hardware design was carried out using Altium Designer.

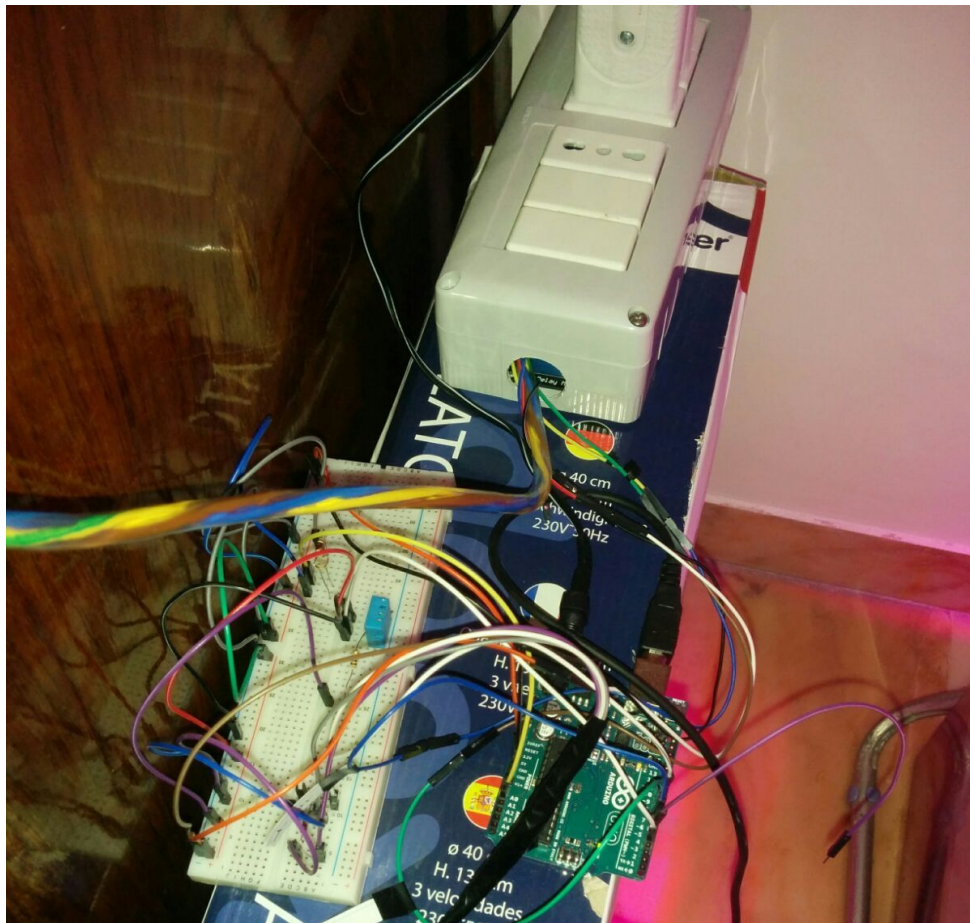
### **Pictures of early system prototype**



*Figure 20: Ebb and Flow system design, clay pebbles are used as substrate*



**Figure 21:** Drain and overflow tube

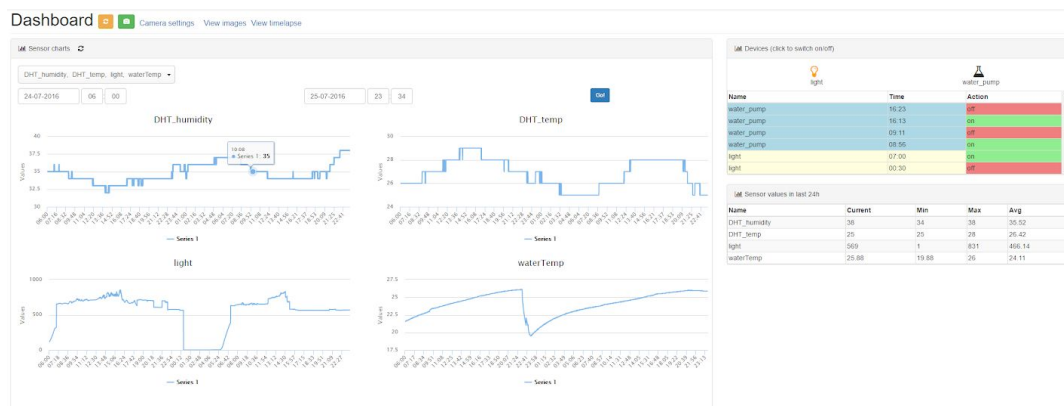


**Figure 22:** Sensors (attached to the breadboard), relays are embedded into the extension cord and used for actuating the light and the pump





**Figure 23: Basil and mint**



**Figure 24: Web app dashboard page**

## Configuration

**Devices**

**light**

ON: 07:00

OFF: 00:30

[Add new interval](#)

**water\_pump**

ON: 08:00, 15:00, 22:00

OFF: 06:15, 15:15, 22:10

[Add new interval](#)

**Sensors**

Update interval (in seconds): 60

Monitor interval (in minutes): 60

**waterTemp**

Min value: 18

Max value: 26

**DHT\_humidity**

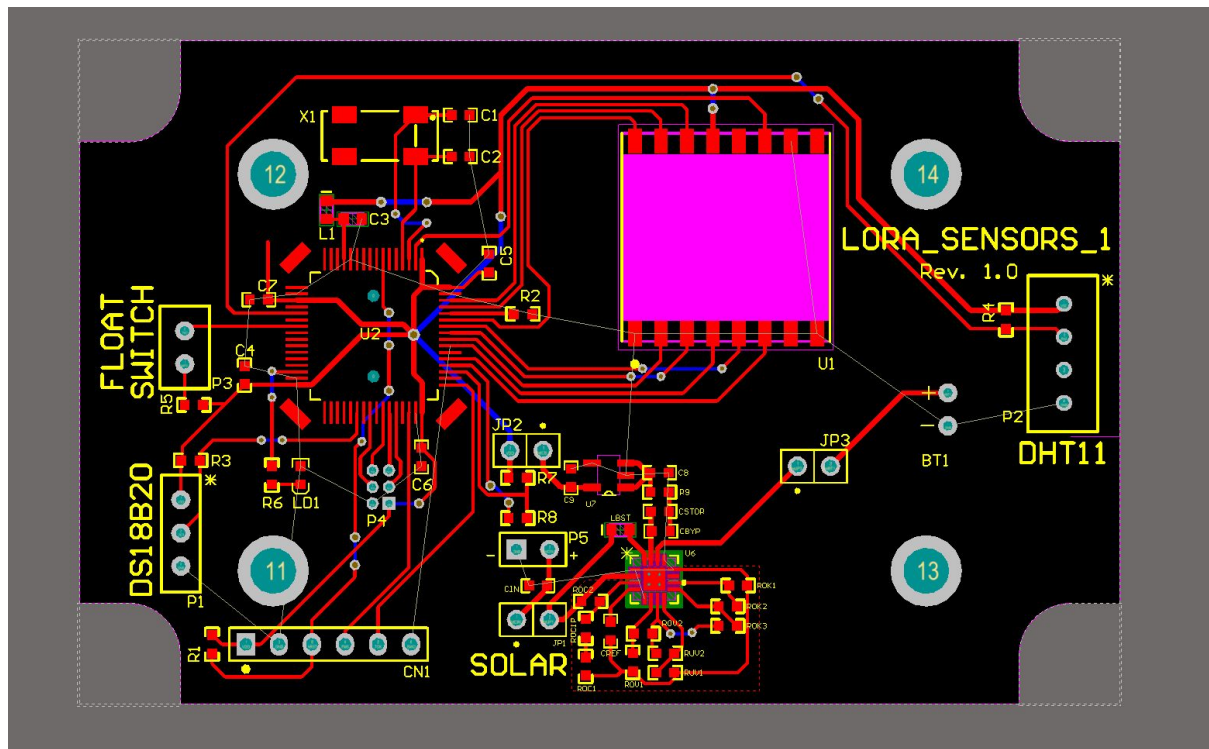
Min value: 33

Max value: 60

[Add new sensor value range](#)

**Figure 25: Web app configuration page**

## 4.1 LoRa\_sensors



*Figure 26: LoRa\_sensors final PCB design. [Altium Designer]*

Apart from the sensors used (more in *Section 4.3*) and the previously described MCU and LoRa module, a small solar cell was added along with the Texas Instruments energy harvester module BQ25504. This module enables extracting power from low-voltage harvesters, requiring only 330 mV to start operating and continuing to operate as low as 80 mV. The electricity generated by the solar cell is used for charging the attached LiPo battery, the energy harvester module also provides constant power to the rest of the system. Since the board was designed to be very low-power, the solar cell should provide enough electricity to keep it running continuously even in artificial light.

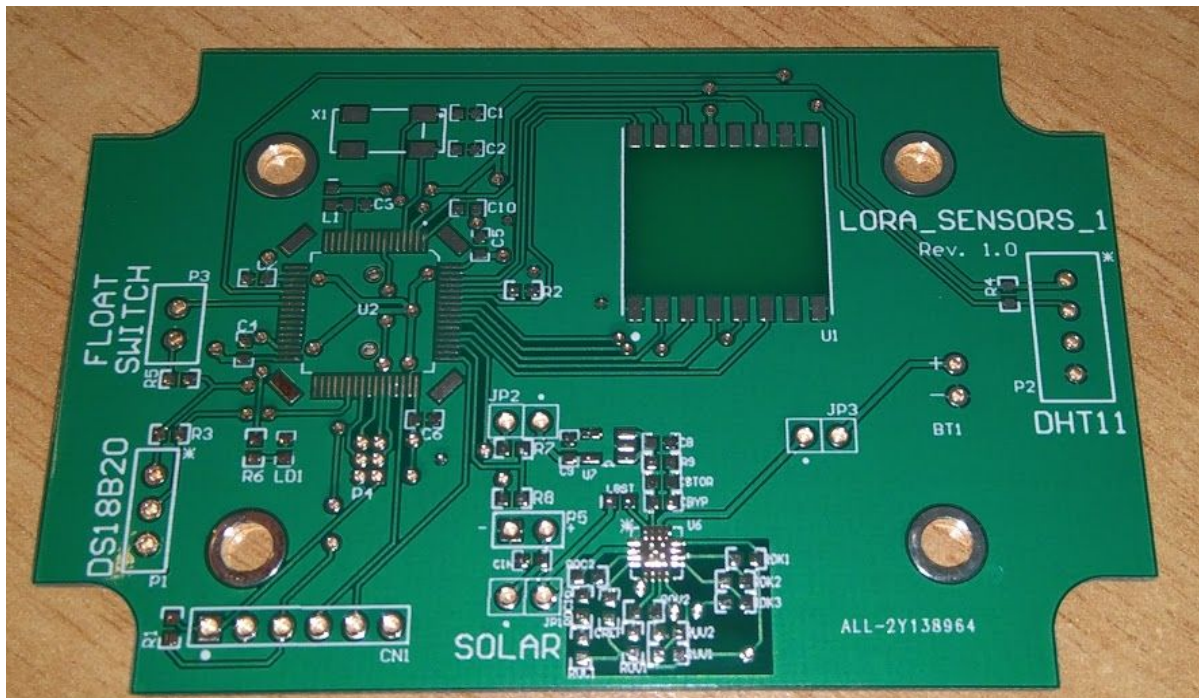


Figure 27: LoRa\_sensors empty PCB

## 4.2 LoRa\_relays

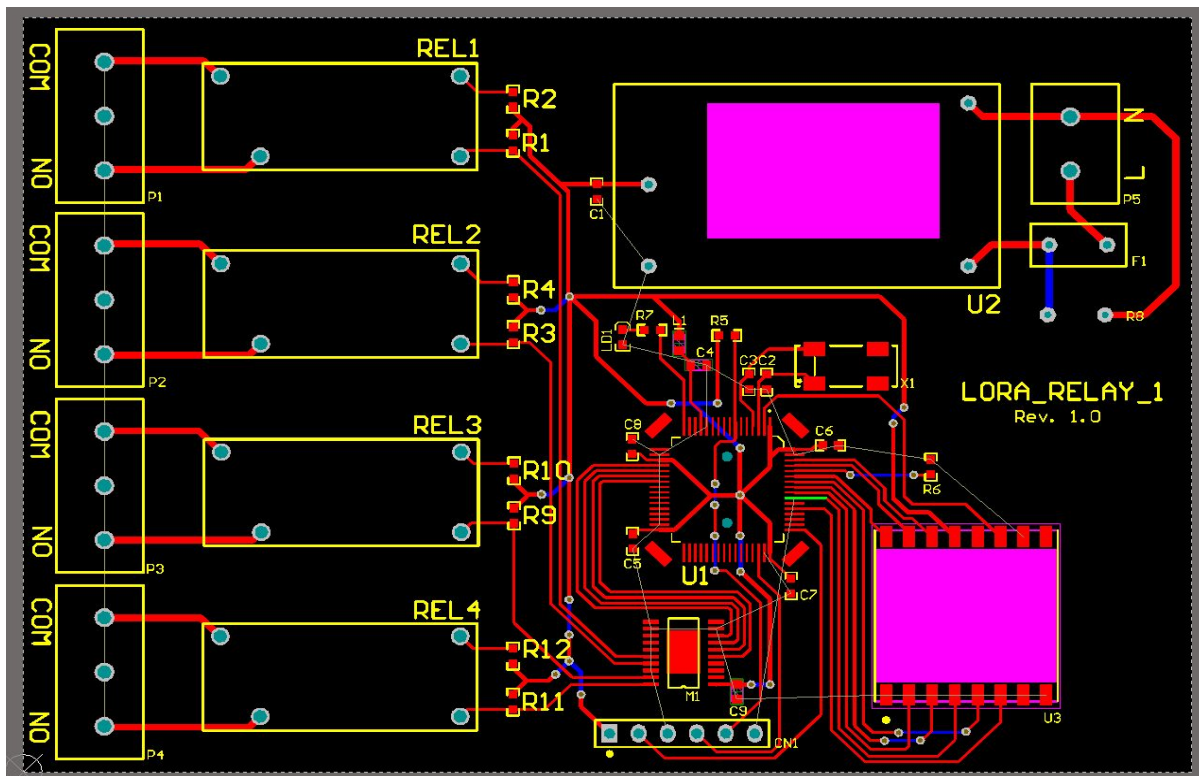
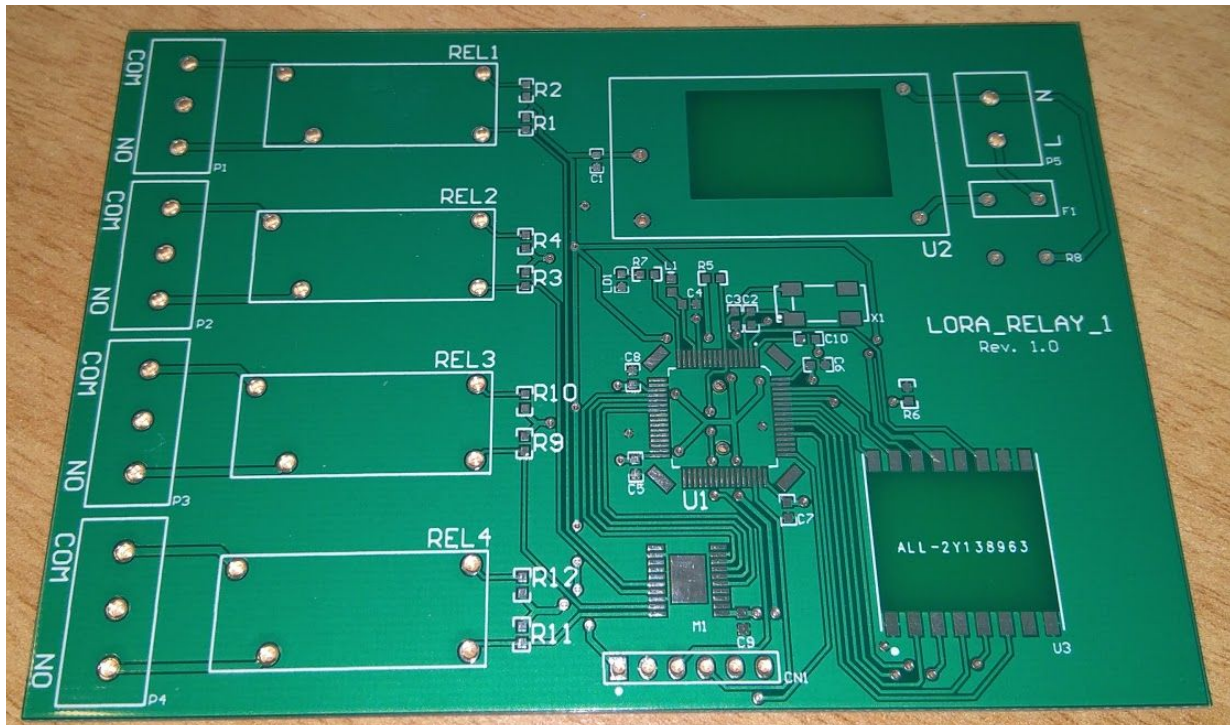


Figure 28: LoRa\_relays final PCB design. [Altium Designer]



The module shares the same MCU and LoRa radio as the LoRa\_sensors. It is powered from the mains, using an AC/DC converter to provide 3.3V to the system. The MCU controls 4 single coil latching relays over the MAX4821 relay driver.



**Figure 29:** LoRa\_relays empty PCB

## 4.3 Hardware

### 4.3.1 DS18B20 Waterproof temperature sensor



**Figure 30: DS18B20**

The DS18B20 digital thermometer provides 9-bit to 12-bit Celsius temperature measurements and has an alarm function with nonvolatile user-programmable upper and lower trigger points. The DS18B20 communicates over a 1-Wire bus that requires only one data line (and ground) for communication with a central microprocessor. In addition, the DS18B20 can derive power directly from the data line (“parasite power”), eliminating the need for an external power supply. Each DS18B20 has a unique 64-bit serial code, which allows multiple DS18B20s to function on the same 1-Wire bus. Thus, it is simple to use one microprocessor to control many DS18B20s (up to 128).

#### Features

- Temperature range -55°C to +125°C
- $\pm 0.5^{\circ}\text{C}$  Accuracy from -10°C to +85°C
- Parasitic power mode requires only 2 pins for operation (Data and Ground)
- Flexible user-definable nonvolatile alarm settings with alarm search command identifies devices with temperatures outside programmed limits

#### 1-Wire protocol

1-Wire is a device communications bus system that provides low-speed data transfer, signaling and power over a single conductor. It is based on a similar concept like I2C but with lower data rates and longer wire range. A distinctive feature is that the bus needs only two wires in order to work, data and ground. To use this feature, 1-Wire devices have an embedded capacitor to store the charge which is used to power the device during the periods when the data line is active.

In 1-Wire many devices can share the same bus, there is only one master that initiates activity on the bus. Each device has a unique 64-bit serial number, the least significant byte tells the type of the device, while the most significant byte is an 8-bit CRC.

There are several standard commands for broadcasting, along with the commands used to address specific device.

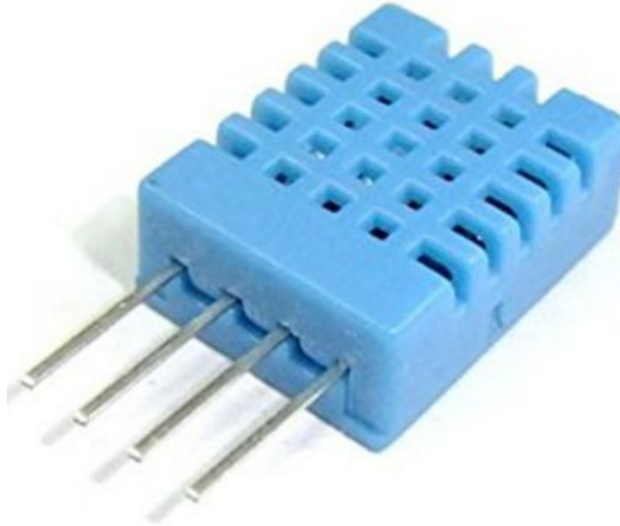
The 1-Wire network is implemented as an open drain master device connected to one or more open drain slaves. There is a single pull-up resistor that is common to all devices and serves to pull up the bus to 3 or 5 volts, also it can provide the power to the slave devices.

## **Communication**

Communication starts when either a master or slave pull the bus to low. The master starts a transmission by sending a *reset* pulse, which is pulling the line to 0 for at least 480  $\mu$ s. After this, all devices on the bus will respond with a *presence* pulse, that is pulling the line to low for at least 60  $\mu$ s. Sending the actual data is performed by pulling the line either 1-15  $\mu$ s corresponding to a "1", or 60  $\mu$ s for a "0".

The basic sequence of commands is the reset pulse followed by an 8-bit command, then the data is transferred in chunks of 8-bits. Errors can be detected with an 8-bit CRC.

### 4.3.2 DHT11 Temperature/Humidity sensor



**Figure 31: DHT11**

The DHT11 is a basic, ultra low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and outputs a digital signal on the data pin (no analog input pins needed).

#### **Features**

- 3 to 5V power and I/O
- 2.5mA max current use during conversion (while requesting data)
- Good for 20-80% humidity readings with 5% accuracy
- Good for 0-50°C temperature readings  $\pm 2^\circ\text{C}$  accuracy
- No more than 1 Hz sampling rate (once every second)

#### **Overall Communication process**

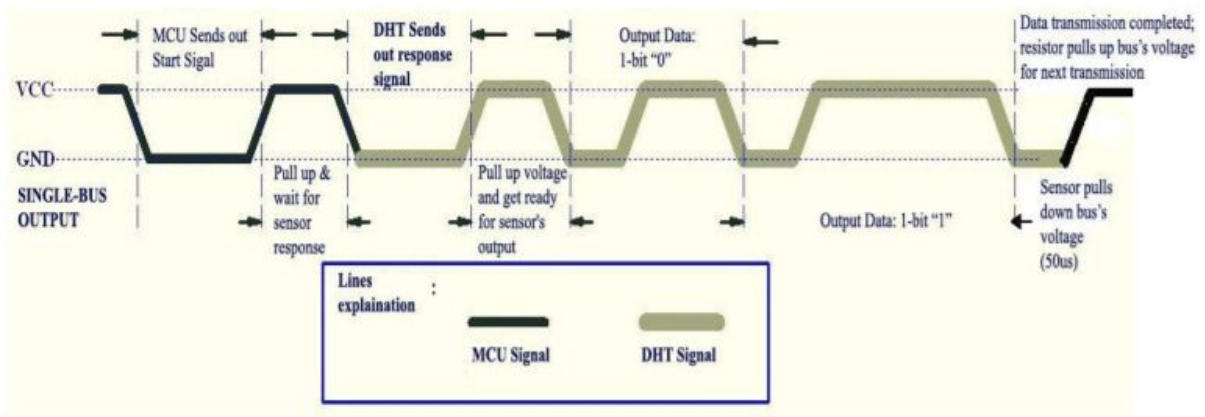


Figure 32: MCU initiates communication

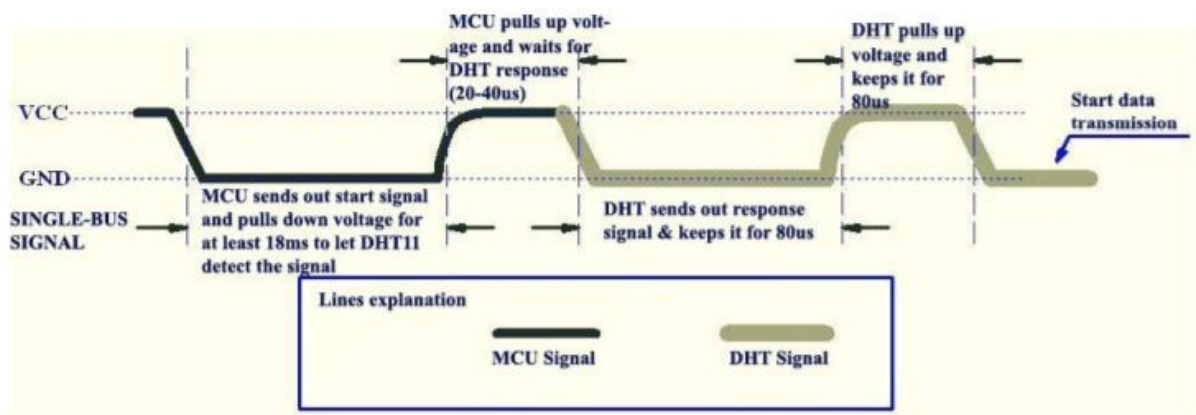


Figure 33: MCU sends a start signal

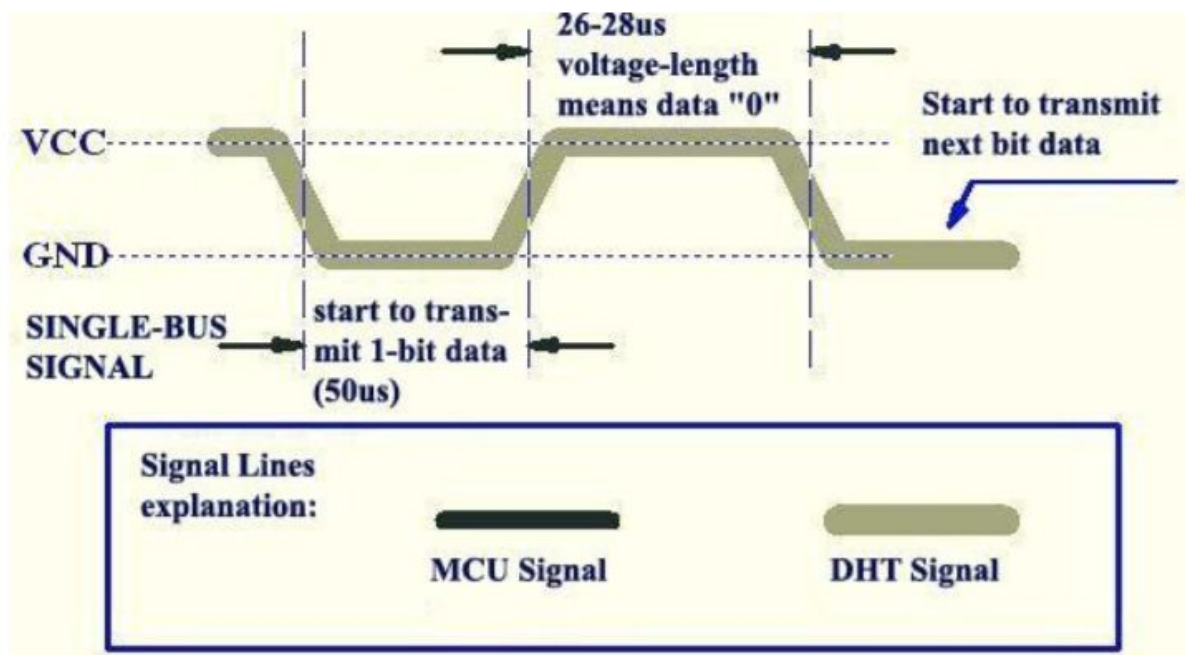
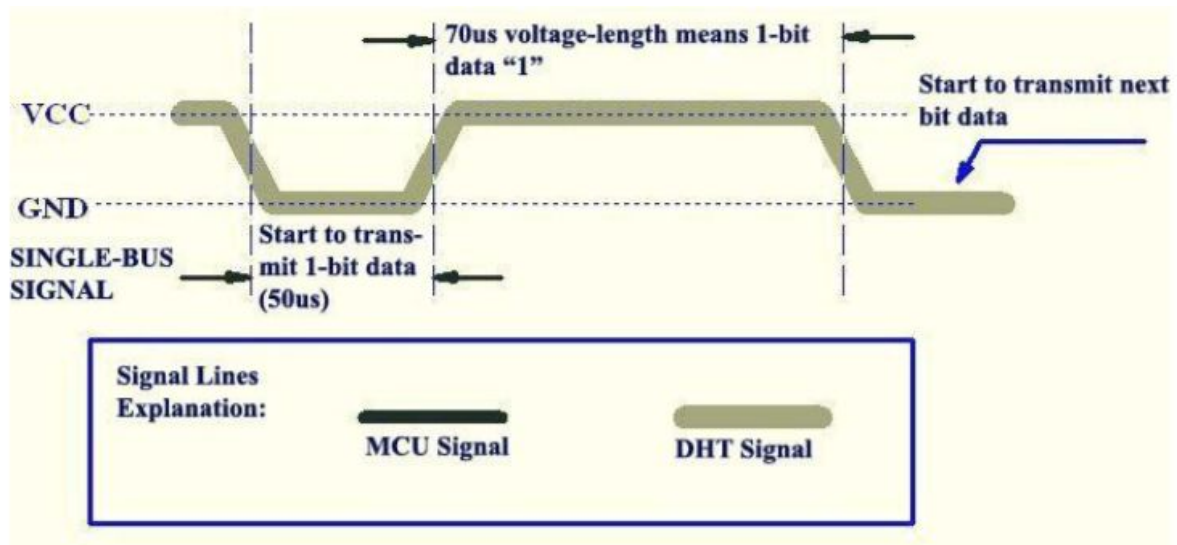


Figure 34: DHT responds with a "0"





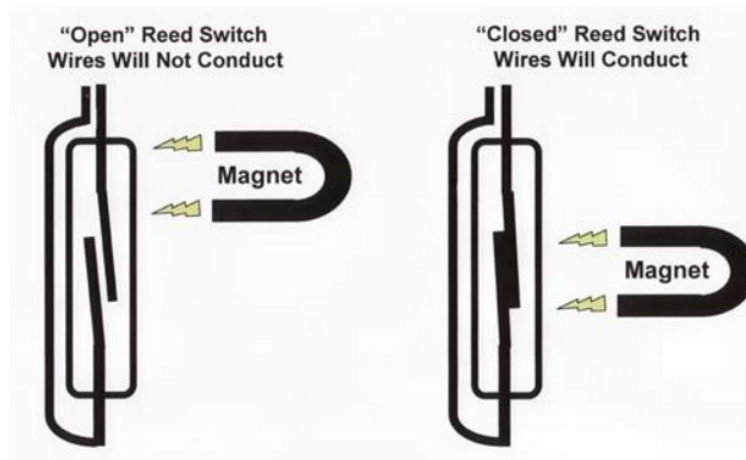
**Figure 35:** DHT responds with a "1"

#### 4.3.3 FD45PI10 magnetic float switch



**Figure 36:** Float switch

This device operates on the principle of a Reed switch, which is basically a pair of thin metal wires (reeds) that have a small gap between them, when a magnet is put close to them they touch and close the circuit.



*Figure 37: Reed switch operating illustration*

#### 4.3.4 RFM96W

RFM96W is a LoRa transceiver produced by HopeRF and based on the Semtech SX1276 chip. It uses SPI communication interface with ~100mA peak current during +20dBm transmit, ~30mA during active radio listening.

### 4.4 Software

#### 4.4.1 Node-Red

Node-Red is a development tool that utilizes a flow programming style, used for connecting devices, APIs and online services that together make Internet Of Things. It provides a browser-based editor where nodes, representing various elements and services, are wired together to make flows which can be easily deployed to runtime.

#### 4.4.2 InfluxDB

InfluxDB is an open-source time series database developed by InfluxData. It is written in Go and optimized for fast, high-availability storage and retrieval of time series data in fields such as operations monitoring, application metrics, Internet of Things sensor data, and real-time analytics.

InfluxDB has no external dependencies and provides an SQL-like language with built in time-centric functions for querying a data structure composed of measurements, series, and points. Each point consists of several key-value pairs called the fieldset and a timestamp. When grouped together by a set of key-value pairs called the tagset, these define a series. Finally, series are grouped together by a string identifier to form a measurement.

Values can be 64-bit integers, 64-bit floating points, strings, and booleans.

Points are indexed by their time and tagset.

## 5 Conclusion

Recent advances in cyber-physical systems, low-power wireless networks and the Internet Of Things coupled with cloud and cognitive computing provide a promising solution to automating nearly all fields of industry, ranging from agriculture, manufacturing, energy, mining to other segments like healthcare, smart cities, transportation and home automation. This will lead to a revolution in all these fields, commonly called “Industry 4.0”, massively improving efficiency, cutting down costs and providing new ways of development that weren’t possible before. These changes are not going to come all at once but steps are already being taken, with companies around the world deploying the systems and seeing tangible benefits of using them.

## 6 References

- [1] <https://www.elprocus.com/introduction-to-wireless-sensor-networks-types-and-applications/>
- [2] [https://en.wikipedia.org/wiki/Wireless\\_sensor\\_network](https://en.wikipedia.org/wiki/Wireless_sensor_network)
- [3] <https://opensource.com/resources/what-open-hardware>
- [4] [https://www.dropbox.com/sh/7er2x1u702rywmj/AACnrS-gPlptVFXPU-Vcxmyla?dl=0&preview=TUeV\\_Rheinland\\_Overview\\_LoRa\\_and\\_LoRaWANtmp.pdf](https://www.dropbox.com/sh/7er2x1u702rywmj/AACnrS-gPlptVFXPU-Vcxmyla?dl=0&preview=TUeV_Rheinland_Overview_LoRa_and_LoRaWANtmp.pdf)
- [5] <https://www.digikey.com/articles/techzone/2016/nov/lorawan-part-1-15-km-wireless-10-year-battery-life-iot>
- [6] <https://en.wikipedia.org/wiki/Hydroponics>
- [7] <https://www.thethingsnetwork.org/docs/network/architecture.html>
- [8] <https://www.thethingsnetwork.org/docs/lorawan/security.html>
- [9] Ramon Sanchez-Iborra, Jesus Sanchez-Gomez, Juan Ballesta-Viñas, Maria-Dolores Cano and Antonio F. Skarmeta. Performance Evaluation of LoRa Considering Scenario Conditions. *Sensors* **2018**, 18(3), 772.
- [10] [http://www.st.com/content/st\\_com/en/products/microcontrollers/stm32-32-bit-arm-cortex-mcus/stm32-ultra-low-power-mcus/stm32l4-series/stm32l4x6/stm32l476rg.html](http://www.st.com/content/st_com/en/products/microcontrollers/stm32-32-bit-arm-cortex-mcus/stm32-ultra-low-power-mcus/stm32l4-series/stm32l4x6/stm32l476rg.html)
- [11] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7):1645 – 1660, 2013.
- [12] Ioannis Chatzigiannakis, Tassos Dimitriou, Sotiris E. Nikolettseas, and Paul G. Spirakis. A probabilistic algorithm for efficient and robust data propagation in wireless sensor networks. *Ad Hoc Networks*, 4(5):621–635, 2006.
- [13] Ioannis Chatzigiannakis, Georgios Mylonas, and Sotiris E. Nikolettseas. Modeling and evaluation of the effect of obstacles on the performance of wireless sensor networks. In *Proceedings 39th Annual Simulation Symposium (ANSS-39 2006)*, 2-6 April 2006, Huntsville, Alabama, USA, pages 50–60. IEEE Computer Society, 2006.

- [14] Ioannis Chatzigiannakis, Elisavet Konstantinou, Vasiliki Liagkou, and Paul G. Spirakis. Design, analysis and performance evaluation of group key establishment in wireless sensor networks. *Electr. Notes Theor. Comput. Sci.*, 171(1):17–31, 2007.
- [15] Tobias Baumgartner, Ioannis Chatzigiannakis, Sándor P. Fekete, Stefan Fischer, Christos Koninis, Alexander Kröller, Daniela Krüger, Georgios Mylonas, and Dennis Pfisterer. Distributed algorithm engineering for networks of tiny artifacts. *Computer Science Review*, 5(1):85–102, 2011.
- [16] Ioannis Chatzigiannakis, Athanasios Kinalis, and Sotiris E. Nikolettseas. An adaptive power conservation scheme for heterogeneous wireless sensor networks with node redeployment. In *SPAA 2005: Proceedings of the 17th Annual ACM Symposium on Parallelism in Algorithms and Architectures*, July 18-20, 2005, Las Vegas, Nevada, USA, pages 96–105, 2005.
- [17] Dimitrios Amaxilatis, Ioannis Chatzigiannakis, Shlomi Dolev, Christos Koninis, Apostolos Pyrgelis, and Paul G. Spirakis. Adaptive hierarchical network structures for wireless sensor networks. In *Ad Hoc Networks - Third International ICST Conference, ADHOCNETS 2011*, Paris, France, September 21-23, 2011, Revised Selected Papers, volume 89 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 65–80. Springer, 2011.
- [18] Ioannis Chatzigiannakis, Athanasios Kinalis, and Sotiris E. Nikolettseas. Efficient data propagation strategies in wireless sensor networks using a single mobile sink. *Computer Communications*, 31(5):896–914, 2008.
- [19] Luis Sanchez, Luis Muoz, Jose Antonio Galache, Pablo Sotres, Juan R. Santana, Veronica Gutierrez, Rajiv Ramdhany, Alex Gluhak, Srdjan Krco, Evangelos Theodoridis, and Dennis Pfisterer. Smartsantander: lot experimentation over a smart city testbed. *Computer Networks*, 61(0):217 – 238, 2014.
- [20] Ioannis Chatzigiannakis, Andrea Vitaletti, and Apostolos Pyrgelis. A privacy-preserving smart parking system using an iot elliptic curve based security platform. *Computer Communications*, 89-90:165–177, 2016.
- [21] M. Centenaro, L. Vangelista, A. Zanella, and M. Zorzi. Long-Range Communications in Unlicensed Bands: the Rising Stars in the IoT and Smart City Scenarios. *IEEE Wireless Communications*, 23, October 2016.
- [22] U. Raza, P. Kulkarni, and M. Sooriyabandara. Low power wide area networks: An overview. *IEEE Communications Surveys Tutorials*, (99):1 – 1, 2017.
- [23] Alexandros-Apostolos A. Boulogeorgos, Panagiotis D. Diamantoulakis, and George K. Karagiannidis. Low power wide area networks (lpwans) for internet of things (iot) applications: Research challenges and future trends. *CoRR*, abs/1611.07449, 2016.
- [24] Dimitrios Amaxilatis, Orestis Akrivopoulos, Ioannis Chatzigiannakis, and Christos Tselios. Enabling stream processing for people-centric iot based on the fog computing paradigm. In *22nd IEEE International Conference on Emerging Technologies and Factory*

Automation, ETFA 2017, Limassol, Cyprus, September 12-15, 2017, pages 1–8. IEEE, 2017.

[25] <https://www.iso.org/news/Ref2183.htm>